WHAT IS CLAIMED IS:

1. A processing system that executes multiple instruction contexts comprising:

an instruction memory for storing instructions that are executed by the system;

a processor unit for executing instructions in a pipelined fashion;

a plurality of context registers for storing instructions and instruction addresses for contexts awaiting execution; and

fetch logic for selecting an address from one of said plurality of context registers and for selecting an instruction from a second of said plurality of context registers for execution by said processor unit, with the fetch logic selecting the address and the instruction substantially simultaneously for each cycle of execution of said processor unit.

2. The system of claim 1 wherein said processor unit outputs control information related to the execution of an instruction, and said fetch logic selects a third of said plurality of context registers for input of the control

information substantially simultaneously with the selection of the address and the instruction in each cycle of execution of said processor unit.

3. The system of claim 2 wherein the fetch logic selects a different one of said plurality context registers in a round robin manner during each corresponding, successive cycle of execution of said processor.

4. The system of claim 3 wherein the control information includes an indication that a branch instruction was executed by a previous context and the control information also including an instruction address of a branch taken.

5. The system of claim 3 wherein the control information includes an indication that a branch instruction was executed by a previous context and the control information also including an instruction address of a branch not taken.

6. The system of claim 3 wherein the control information includes an indication that a subroutine instruction was executed by a previous context and the control information includes an instruction address of subroutine.

7. The system of claim 4 further comprises:

scheduling logic that schedules execution of the contexts by storing an instruction address for a context ready for execution in an available one of said plurality of context registers.

8. The system of claim 7 wherein the control information is also input to said scheduling logic, the control information including an indication that a context exit instruction was executed by a first context.

9. The system of claim 8 wherein said scheduling logic stores a context instruction address for a second context into an available one of said plurality of context registers based on the indication that a context exit instruction was executed by the first context.

10. A method of operating a processing system, the method comprising:

scheduling a plurality of contexts to be executed by said system, said scheduling comprises:

storing an instruction address for each of said plurality of contexts in a corresponding one of a plurality of context registers;

selecting a first instruction address and a first instruction from a first and second of said plurality of context registers in a first cycle of execution of the system;

selecting a second instruction address from the second of said plurality of registers and a second instruction from a third of said plurality of registers in a second cycle of execution of the system.

11. The method of claim 10 further comprises:

storing control information in one of said plurality of context registers in each cycle of execution of the system.

12. The method of claim 11 further comprises:

determining a branch taken instruction address based on the execution of a previous instruction by the system; and

storing the branch taken address in one of the plurality of context registers.

13. The method of claim 11 further comprises:

determining a branch not taken instruction address based on the execution of a previous instruction by the system; and

storing the branch not taken instruction address in one of the plurality of context registers.

14. The method of claim 11 further comprises:

determining a subroutine instruction address based on the execution of a previous instruction by said system; and

storing the subroutine address in one the plurality of context registers.

15. The method of claim 11 wherein said scheduling further comprises:

determining that a context exit instruction was executed from a first one of said context registers;

storing an instruction address for a new context in the first one of said context registers.

16. A computer program stored in a computer readable medium having instructions causing a computer that executes multiple contexts to:

store an instruction address in each of a plurality of context registers;

load a first instruction corresponding to a first instruction address stored in one of the plurality of context registers;

select the first instruction for execution in a first cycle of execution of said computer; and

load a second instruction corresponding to a second instruction address stored in a second of the plurality of context registers substantially simultaneously with the selection of the first instruction.

17. The computer program of claim 16 further comprising instruction causing a computer that executes multiple contexts to:

determine control information related to the execution of a previous instruction in each cycle of execution of the computer; and

store the control information in a one of the plurality of context registers substantially simultaneously with the selection of the first instruction.

18. The computer program of claim 17 further comprising instructions causing a computer that executes multiple contexts to:

determine whether a branch is taken as the result of the execution of the previous instruction; and

store a branch taken instruction address in one of the plurality of context registers.

19. The computer program of claim 17 further comprising instructions causing a computer that executes multiple contexts to:

determine whether a branch is not taken as the result of the execution of the previous instruction; and

store a branch not taken instruction address in one of the plurality of context registers.

20. The computer program of claim 17 further comprising instructions causing a computer that executes multiple contexts to:

determine a subroutine address as the result of the execution of the previous instruction; and

store the subroutine address in one of the plurality of context registers.

21. The computer program of claim 17 further comprising instructions causing a computer that executes multiple

contexts to:

determine whether the previous instruction from a first one of the plurality of context registers was a context exit instruction; and

store a new context instruction address in the first one of the plurality of context registers.